Optimizing a Fraud Detection Process[1]

Nuno Homem
TULisbon – Instituto Superior Tecnico
INESC-ID
R. Alves Redol 9, 1000-029 Lisboa
Portugal
nuno_homem@hotmail.com

Joao Paulo Carvalho
TULisbon – Instituto Superior Tecnico
INESC-ID
R. Alves Redol 9, 1000-029 Lisboa
Portugal
joao.carvalho@inesc-id.pt

Abstract

Fraud in telecommunications services or financial transactions is a major problem as it impacts from 1% to 3% of the revenues. This is in most cases a customer specific behavior that companies need to detect in order to minimize it. Detecting specific types of behavior as soon as it happens is critical, and to that purpose companies deploy sophisticated detection systems. The biggest challenge to fraud detection systems is accurately predict in near real time that a customer is a fraudster or that is service is being used in fraudulent way. As this may happen to any customer at any time it is mandatory to monitor the entire customer base – sometimes several million customers making several transactions per day. Optimizing the detection process is therefore critical. To model and analyze the problem Finite State Automata and Markov Chains were used. To solve the optimization problem Dynamic Programming and Stochastic Hill Climber algorithms were chosen.

Keywords: Fraud, telecommunications, detection, optimization.

1 Introduction

This paper proposes a new method for a fraud detection process in telecommunications. Most fraud systems have two distinct processes, continuous data integration and aggregation process and a fraud detection process. In the first process data from several systems has to be integrated and correlated. This data is generated by N network platforms that have to be transferred into a central platform for processing. This happens as soon as data is available.

In the data integration process, for each customer a set of ready to use data is generated and stored. This is an incremental process that glues customer demographic and state, customer history data (e.g. how long he's been a customer, debt default incidents, historical consumptions), usage data (e.g. average usage for several periods, types of usage, geography of usage – includes both long term and very short term views) and sometimes more complex indicators (e.g. traffic fingerprints, transaction dispersion. Detailed transaction and event data is also stored to allow for future detailed analysis.

The fraud detection process can be repeated at pre-defined intervals or as soon as a previous round ends. In each round all customers (or at least active customers) should be checked. Checking a customer requires performing a set of tests on the previously generated information. The results of these tests may indicate some probable problems (several types of problem may be detected). These detected problems can then be further analyzed. Usually this second level of testing requires more complex processing, sometimes very complex and detailed analysis of traffic data that may take some time. These tests are usually more accurate but the time it takes to perform them makes it impossible to check every customer in that way. The results of these tests can then dismiss the problem, confirm it (at least assign a stronger probability) or point into another direction that may require further testing. The outcome of the detection process may be a fraud alarm. These alarms will be analyzed by a fraud analyst and may prove to be real or false.

[1] This work was supported in part by the FCT - Portuguese Foundation for Science and Technology under project PTDC/EEA-ELC/66259/2006

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 583–590
Torremolinos (Málaga), June 22–27, 2008

There's a high reward to detection of real alarm, a high cost of not detecting a fraud situation but also a penalty for triggering false alarms as this may lead to loss of revenue or even to the loss of the customer. The alarm accuracy usually takes several days to check.

In this paper the chosen approach to the detection process is distinct from what is traditionally used. In traditional methods the tests are executed following a specific sequence and the decision is taken upon the results of those tests. In this approach we sometimes decide not to do some tests randomly with a specific probability that is obtained from the test quality analysis. This allows the system to be optimized using probabilities of the tests generating true positive and false positive results. In the traditional approach the fact that tests may generate false positive results is left for the final analysis case and the huge numbers of false positives that are generated in those fraud detection systems are one of the main reasons for fraud detection process failure.

2 Concrete case

The following case will only focus on the fraud detection process, assuming data integration is already done. This particular fraud detection system allows 8 types of tests to be performed on customer data. From the 8 available tests, 3 are instantaneous and 5 take some time to complete. Each test has only positive or negative result.

The instantaneous tests use only pre-computed data and are always performed before all other tests. These tests check whether:

- T0 – Is a recent customer? Checks if the customer is not sufficiently known by the operator, no history exists to access is behavior. Unknown customers or recent customers pose increased risks to the operator.
- T1 – High diversity of use? Checks the usage behavior of the customer (the type of services used, calls made, list of destinations). A large diversity of usage or destinations may indicate a non personal use of the services, for instance selling of calls to several persons by the customer without intention of paying for it.
- T2 – High usage? Checks if the customer is making too many calls, especially to high value destinations (international calls, roaming calls). This is a fast check because only the number of calls, total duration and a simple value estimate (based on destination type) is used (this value can be computed as calls arrive).
- The more complex tests, those that require additional data to be used or external systems to be called are:
- T3 – Credit check - Checks whether the credit limit of the customer is the appropriate by using its invoice and payment history (if available), its demographic data and the history of that customer in internal and external lists (sometimes operators share lists of bad customer between themselves).
- T4 – Traffic Pattern – Checks whether the usage is a typical fraudster usage. Fraudsters tend to have very high numbers of calls of very high value, like international destination calls (especially to uncommon destinations), roaming calls, added value services or content usages.
- T5 - Automated caller - Checks if the type of use is typical of automated systems. Unlike humans, machines are capable of doing huge amounts of calls (hundreds) to very diverse destinations. Usually a person makes most of its calls to just a few destinations. Machines don't receive calls. Automated systems are used to reroute traffic through less expensive channels (and operators lose the difference) or to resell traffic that is never paid.
- T6 - External agencies – Checks if the customer is in any of the external credit agencies bad payment lists or in the central bank credit denial or credit incidents list. These inquires take a long time and cost money.
- T7 – Check list of Fraudsters? Checks whether the customer has a similar fingerprint to an already known fraudster. These fingerprints are usually obtained by using the call destinations and call usage characteristics and generating a very high dimension vector. This

vector has then to be compared to the fingerprints of known fraudsters.

The detection process applies the 3 initial tests to each customer; this takes only a minimal processing time (a few milliseconds) since data is already computed. Those tests will allow the system to identify potential risks that will be used to decide if additional optional tests are to be made:

- If all 3 tests are negative then no further checking is required and is not a fraudster.
- If T0 is positive then the customer is not well known to the operator – recent customers require a credit analysis – then T3 should be performed.
- If T0 and T1 are positive then the customer might be an over spending customer – credit limits for recent customers should be tighter – then T4 should be performed.
- If T2 is positive then this is potentially a false customer, a machine redirecting traffic – then T5 should be performed.

Thus the system will have to decide whether to make the test or not based not only on the rules but also on a probability. Even if a test should be made there will be a probability of the system deciding not to make it. This may lead to a fraudster not being detected in this detection run. Probably it will be detected later. The rules for performing the last optional tests are:

- T6 should only be performed if T3 is positive.
- T7 should only be performed if T4 is negative.
- T6 or T7 are only evaluated after T3, T4 and T5 complete

Finally, if any of T4, T5, T6 or T7 is positive the customer is classified as a fraudster.

Each test has a failure probability. This means that each test can indicate to a potential fraudster when the customer is OK or say it is OK when it is not. Once the potential fraudsters are identified, an investigation has to be made by fraud analysts. The analysts may take some actions on the customer. Naturally this is quite expensive not only because of the time and resources needed but also because some of actions may result in customer dissatisfaction if applied to OK customers. Therefore, the system must avoid generating false positives.

In this new approach we consider that system should optimize the probabilities of performing each of the optional tests in order to improve its detection rate of fraudsters and reducing false positives. This maximization should take account of the gains and losses of the process.

Over time the results of the analysis by fraud analysts and the behavior of the customer (not paying its debts) will identify real fraudsters. Once a fraudster is detected it will be blocked. If a customer is identified as a fraudster and the analysis clear him, then the customer gets back to the detection system. Figure 1 provides a block diagram of the overall detection process.

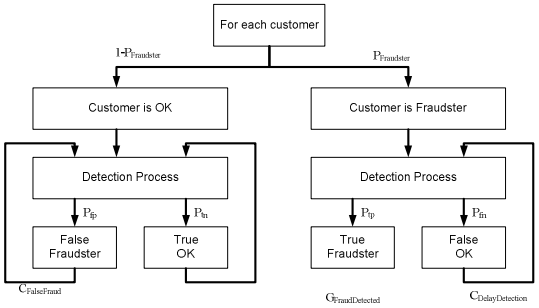

Figure 1: Overall detection process.

Process gains and losses must be considered:

- G_{FD} – the average losses due to a fraudster – the avoided loss when a fraudster is detected.
- C_{FF} – the average cost of analyzing a false positive and the potential cost of customer dissatisfaction.
- C_{DD} – the cost of missing a fraudster in the run.

The following will also be considered:

- P_F – probability that a customer is a fraudster.
- P_{fn} – probability of detecting false negatives.
- P_{fp} – probability of detecting false positives.
- P_{tp} – probability of detecting true positives.

The system decides the probabilities of making tests T3, T4, T5, T6 and T7 (pT3, pT4, pT5, pT6 and pT7 respectively). Detection process is repeated for each customer every day.

3 Detection system representation

The detection system can be represented by a Finite State Automata (FSA) [2] [1]. Since the tests in can be grouped, the states can represent the several tests being performed at the same time. For example, T34 means that T3 and T4 can be done simultaneously. Figure 2 represents the FSA of the detection algorithm. The FSA evolves from top to bottom in sequence.

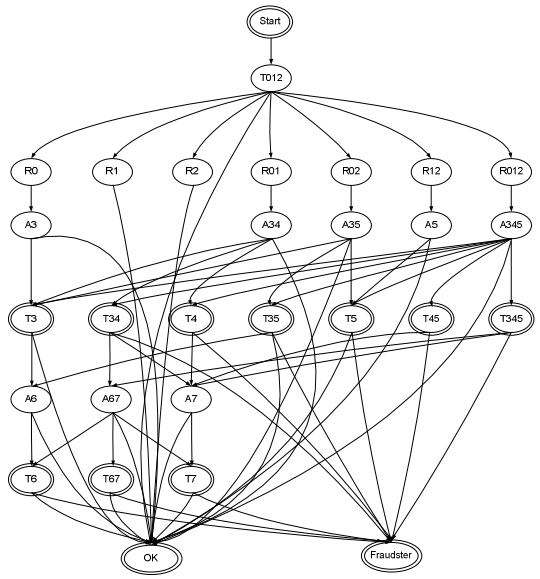

Figure 2: Detection algorithm FSA.

Marked states are tangible states, unmarked states are vanishing states (only immediate transitions enabled):

- States labeled T represents test performed.
- States labeled R represents results of the initial tests.
- States labeled A represents choices of complex tests to perform (before randomly selecting which are to be in reality made).
- The OK state means that a customer is not a fraudster and the Fraudster state identifies the customer as a fraudster.

Let T_j be the set of tests. Consider p_j as the probability of test T_j giving a positive result. We will consider all tests independent, i.e., not correlated.

From this FSA and by applying the transition probabilities to each link we can obtain the corresponding Markov Chain (MC)[2]. This MC can be further simplified by eliminating the transient states. Figure 3 shows the simplified Detection FSA.

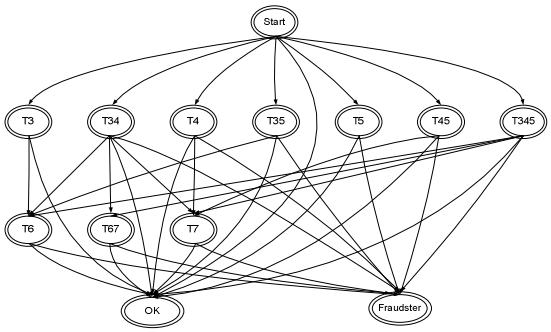

Figure 3: Simplified detection FSA.

The probabilities and times for each transition (L) in this MC are show in Table 1.

Table 1: Transition probabilities table.

L		p_L	
Initial State	**Final State**	**Probability**	**= fx**
Start	OK	(1-p0).(1-p1).(1-p2)+(1-p0).p1.(1-p2)+(1-p0).(1-p1).p2+(1-pT3). p0.(1-p1).(1-p2)+(1-pT3).(1-pT4). p0.p1.(1-p2)+(1-pT3).(1-pT5). p0.(1-p1).p2+(1-pT5). (1-p0).p1.p2+(1-pT3).(1-pT4).(1-pT5). p0.p1.p2	f1
Start	T3	pT3. p0.(1-p1).(1-p2)+pT3.(1-pT4). p0.p1.(1-p2)+pT3.(1-pT5). p0.(1-p1).p2+pT3.(1-pT4).(1-pT5). p0.p1.p2	f2
Start	T34	pT3.pT4. p0.p1.(1-p2)+pT3.pT4.(1-pT5). p0.p1.p2	f3
Start	T345	pT3.pT4.pT5. p0.p1.p2	f4
Start	T35	pT3.pT5. p0.(1-p1).p2+pT3.(1-pT4).pT5. p0.p1.p2	f5
Start	T4	(1-pT3).pT4. p0.p1.(1-p2)+(1-pT3).pT4.(1-pT5). p0.p1.p2	f6
Start	T45	(1-pT3).pT4.pT5. p0.p1.p2	f7
Start	T5	(1-pT3).pT5. p0.(1-p1).p2+pT5. (1-p0).p1.p2+(1-pT3).(1-pT4).pT5. p0.p1.p2	f8
T3	OK	(1-p3)+(1-pT6). p3	f9
T3	T6	pT6. p3	f10
T34	Fraudster	p4	f11
T34	OK	(1-pT7). (1-p4).(1-p3)+(1-pT6).(1-pT7). (1-p4).p3	f12
T34	T6	pT6.(1-pT7). (1-p4).p3	f13
T34	T67	pT6.pT7. (1-p4).p3	f14
T34	T7	pT7. (1-p4).(1-p3)+pT7.(1-pT6). (1-p4).p3	f15
T345	Fraudster	p4+p5-p4.p5	f16

T345	OK	(1-pT7). (1-p3).(1-p4).(1-p5)+(1-pT6).(1-pT7). p3.(1-p4).(1-p5)	f17
T345	T67	pT6.pT7. p3.(1-p4).(1-p5)	f19
T345	T6	pT6.(1-pT7). p3.(1-p4).(1-p5)	f18
T345	T7	pT7. (1-p3).(1-p4).(1-p5)+pT7.(1-pT6). p3.(1-p4).(1-p5)	f20
T4	Fraudster	p4	f24
T4	OK	(1-pT7). (1-p4)	f25
T4	T7	pT7. (1-p4)	f26
T45	Fraudster	p4+p5-p4.p5	f27
T45	OK	(1-pT7). (1-p4-p5+p4.p5)	f28
T45	T7	pT7. (1-p4-p5+p4.p5)	f29
T5	Fraudster	p5	f30
T5	OK	(1-p5)	f31
T6	Fraudster	p6	f32
T6	OK	(1-p6)	f33
T67	Fraudster	p6+p7-p6.p7	f34
T67	OK	(1-p6).(1-p7)	f35
T7	Fraudster	p7	f36
T7	OK	(1-p7)	f37

4 The full system representation

The full system includes the detection component within the overall process. The system encompasses the two distinct situations, the customer being or not a fraudster. The complete system can also be represented by a FSA (Figure 4).

Using the simplified detection FSA to create the full MC we obtain the transition probabilities shown in Table 2.

Table 2: Reduced transition probabilities table.

L		p_L
Initial State	**Final State**	**Probability**
Start	Fraudster	f3.f11+f4.f16+f5.f21+f6.f24+f7.f27+f8.f30 + f32.(f2.f10+f3.f13+f4.f18+f5.f23)+ f34.(f3.f14+f4.f19)+ f36.(f3.f15+f4.f20+f6.f26+f7.f29)
Start	OK	f1+f2.f9+f3.f12+f4.f17+f5.f22+f6.f25+f7.f28+f8.f31+ f33.(f2.f10+f3.f13+f4.f18+f5.f23)+ f35.(f3.f14+f4.f19)+ f37.(f3.f15+f4.f20+f6.f26+f7.f29)

This model can be used to solve the optimization problem as it can be mapped directly into a Discrete Time Markov Chain (DTMC) in which:

- t_0 – day 0
- t_1 – beginning of the daily process, day 0
- t_2 – daily process completed, day 0
- t_{2N+1} – beginning of the daily process for day N
- t_{2N+2} – daily process completed for day N

We can consider that once the system reaches the True Fraudster state no further evolution is required.

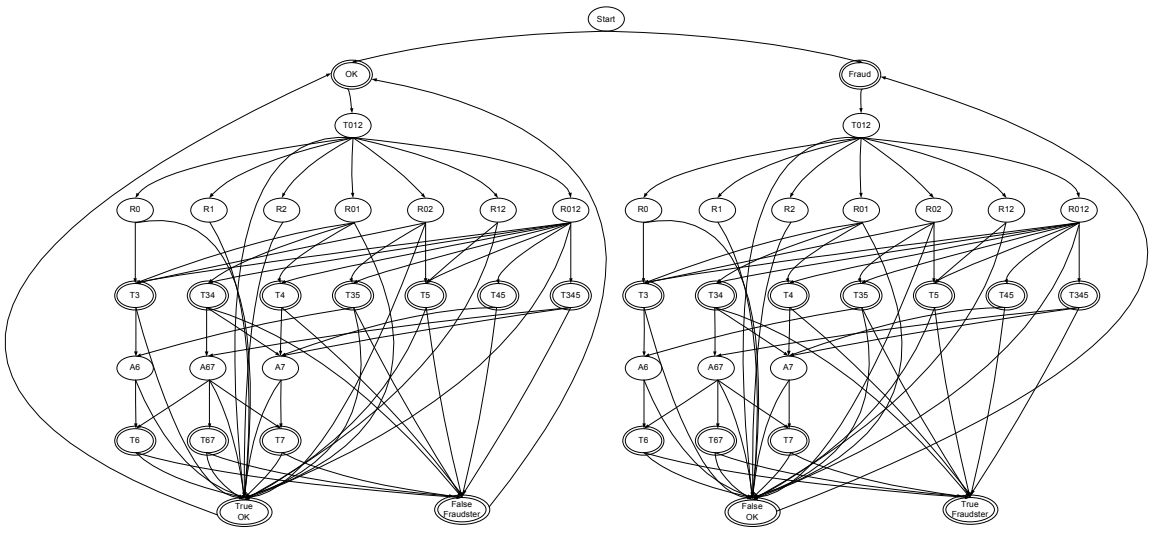

Figure 4: Complete FSA model.

5 Solving this problem

To solve this problem an optimization algorithm is required to find the best decision probabilities for pT3, pT4, pT5, pT6 and pT7 in order to achieve best overall cost while ensuring the required validation time. As parameters are stochastic the algorithm will find the best policies to solve the problem instead of a concrete solution. A policy states the advised action for each possible state.

As each test can give wrong results we have to consider two possibilities, the customer being a Fraudster and the customer being OK:

- P(T j is positive | Fraudster) = p_{jtp}
- P(T j is negative | Fraudster) = p_{jfn}
- P(T j is positive | OK) = p_{jfp}
- P(T j is negative | OK) = p_{jtn}

To calculate P_{fn} and P_{tp} we replace p_j by p_{jtp} in the detection system probabilities model. To calculate P_{fp} we replace p_j by p_{jfp} in the detection system probabilities model. The initial transitions from Start will have probability P_F to Fraud and (1- P_F) to OK.

Each of the states in $_{t2N+2}$ can be assigned a cost:

- True OK: 0;
- False Fraudster: C_{FF} – the average cost of analyzing a false positive and the potential cost of customer dissatisfaction. This is accounted every time the system fails an incorrectly classifies the customer as fraudster;
- False OK: C_{DD} – the cost of not detecting a real fraudster in this run. This is accounted every time the system fails to detect a fraudster;
- True Fraudster: G_{FD} – the average losses due to a fraudster – this is a reward and not a cost as it is the gain generated by the system when it detects a fraudster. This is only accounted once.

The actions will be the values we assign to pT3, pT4, pT5, pT6 and pT7.

In order to optimize our decision let us consider the total expected discounted cost over an infinite horizon for a DTMC given the initial state i.

$$V_\pi(i) = E_\pi [\Sigma_k \alpha^k C(X_k, u_k)] \quad (1)$$

We look for the policy π that minimizes cost:

$$V^*_\pi(i) = \min_{u \in \pi} [V_\pi(i)] \quad (2)$$

Where:

- Initial state i = X0;
- Policy π = {u0, u1,…, uk,…} over infinite horizon;
- Discount rate $0 < \alpha < 1$;

Costs are only relevant for instant t2N+2, otherwise are 0. For simplicity sake let us consider the index translation j = 2N+2. When considering only those instants, nothing changes in the system but notation is considerably simplified.

As the system is initially split in two based on whether the customer is a fraudster or not, costs can also be split between costs for those two situations:

$$C(X_k, u_k) = P_F . C(X_k, u_k)| \text{ Fraudster} + (1- P_F). C(X_k, u_k)| \text{ OK} \quad (3)$$

Consider:

$$C_F(X_k, u_k) = C(X_k, u_k)| \text{ Fraudster} \quad (4)$$

$$C_{Ok}(X_k, u_k) = C(X_k, u_k)| \text{ OK} \quad (5)$$

Consider first the cost when the customer is OK. For the first relevant moment cost is thus:

$$C_{Ok}(X_0, u_0) = P_{fp} . C_{FF} \quad (6)$$

For all instants j:

$$C_{Ok}(X_j, u_j) = P_{fp} . C_{FF} \quad (7)$$

For the fraudsters, as no further costs will be considered after the fraudster is detected, the issue is just to compute the cost of not detecting the fraudster until instant j:

$$C_F (X_0, u_0) = P_{tp} . G_{FD} + P_{fn} . C_{DD} \quad (8)$$

For instant 1, costs are only to consider if the fraudster was not detected in instant 0, this happens with probability Pfn:

$$C_F(X_1, u_1) = P_{fn} .(P_{tp} . G_{FD} + P_{fn} . C_{DD}) \quad (9)$$

For all instants j:

$$C_F(X_j, u_j) = P_{fn}^j.(P_{tp}. G_{FD} + P_{fn}. C_{DD}) \quad (10)$$

Now we can rewrite the cost function:

$$V_\pi(i) = E_\pi [\Sigma_j \alpha^j.[P_F.C_F(X_j, u_j)+(1-P_F).C_{Ok}(X_k, u_k)]] \quad (11)$$

$$V_\pi(i) = E_\pi [\Sigma_j \alpha^j .[P_F.P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}. C_{DD}) + (1-P_F).P_{fp}.C_{FF}]] \quad (12)$$

$$V_\pi(i) = \Sigma_j \alpha^j.[P_F. P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}. C_{DD}) + (1- P_F). P_{fp}. C_{FF}]] \quad (13)$$

We can analyze the consequences of reaching a steady state for this system. This can be easily reached by setting pT3, pT4, pT5, pT6 and pT7 to constant values after a given instant M. This means that after that instant Pfn, Ptp and Pfp are also constant over time and we can rewrite the equation as:

$$V_\pi(i) = \Sigma_{j=0..M-1} \alpha^j.[P_F.P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}.C_{DD})+(1-P_F).P_{fp}.C_{FF}] +\Sigma_{j=M..\infty}\alpha^j.[P_F.P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}.C_{DD}) +(1-P_F).P_{fp}.C_{FF}] \quad (14)$$

$$V_\pi(i) = \Sigma_{j=0..M-1} \alpha^j.[P_F.P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}.C_{DD})+(1-P_F).P_{fp}.C_{FF}] +[P_F.(P_{tp}.G_{FD}+P_{fn}.C_{DD})].\Sigma_{j=M..\infty}\alpha^j.P_{fn}^j +[(1-P_F).P_{fp}.C_{FF}].\Sigma_{j=M..\infty}\alpha^j. \quad (15)$$

$$V_\pi(i) = \Sigma_{j=0..M-1} \alpha^j.[P_F.P_{fn}^j.(P_{tp}.G_{FD}+P_{fn}.C_{DD})+(1-P_F).P_{fp}.C_{FF}]] +P_F.(P_{tp}.G_{FD}+P_{fn}.C_{DD}).P_{fn}^M.\alpha^M/(1-\alpha.P_{fn}) +[(1-P_F).P_{fp}.C_{FF}]. \alpha^M/(1-\alpha) \quad (16)$$

This is an interesting expression as it allows dynamic programming [1] to be used to solve the problem. The initial decisions will contribute at each step to the cost, and the remaining decisions (after reaching the steady state) can be computed in a single step.

A extreme situation would be to consider values for pT3, pT4, pT5, pT6 and pT7 constant for all instants. In this case we can rewrite the equation as:

$$V_\pi(i) = \Sigma_j \alpha^j.[P_F.P_{fn}^j.(P_{tp}.G_{FD+} P_{fn}.C_{DD})] + \Sigma_j \alpha^j.[(1-P_F).P_{fp}.C_{FF}]] \quad (17)$$

$$V_\pi(i) = P_F.(P_{tp}.G_{FD} + P_{fn}.C_{DD})/(1-\alpha.P_{fn}) +(1-P_F). P_{fp}.C_{FF} /(1-\alpha) \quad (18)$$

6 Examples

6.1 Single decision problem

As an example let us analyze a case where only one decision has to me made and pT3, pT4, pT5, pT6 and pT7 remain constant for all instants. Consider the probabilities presented in Table 3. for each test.

Table 3: Test probabilities table.

Test	P_{tp}	P_{fp}
T0	0.5	0.2
T1	0.5	0.2
T2	0.5	0.2
T3	0.6	0.15
T4	0.7	0.1
T5	0.65	0.05
T6	0.7	0.05
T7	0.8	0.01

Consider also the following values for the other variables:

- P_F = 0.01
- G_{FD} = -200.0
- C_{DD} = 2.0
- C_{FF} = 10.0
- α = 0.9

Since the solution space is huge (even considering 0.01 steps we could have to check 100^5 solutions) an optimization algorithm has to be used. In this case, due to its simplicity, the Stochastic Hill Climber algorithm was implemented [3]. The results were:

```
Solution pT3 = 1.0
Solution pT4 = 0.0
Solution pT5 = 0.26
Solution pT6 = 1.0
Solution pT7 = 1.0

Ptp = 0.2556,  Pfn = 0.7444
Pfp = 0.0024,  Ptn = 0.9976

Solution Cost = -1.2630
```

Note that since pT4 is 0.0, pT7 is not relevant as this test is not performed. This can be observed in the solutions as the same result can be achieved with any other pT7 result.

The optimization itself takes less than a second to complete.

6.2 Multiple decision problem

In this example we will consider the same parameters, but we assume that three decisions have to be made for days 0, 1 and 2 (which will be the final value).

First we obtain the steady state solution for M = 2. Then we can use dynamic programming [1] to determine the two other decisions for j = 0 and j = 1.

Solution

```
pT3[0] = 1.0, pT3[1] = 1.0,  pT3[2] = 0.94
pT4[0] = 1.0, pT4[1] = 0.11, pT4[2] = 0.0
pT5[0] = 1.0, pT5[1] = 1.0,  pT5[2] = 0.0
pT6[0] = 1.0, pT6[1] = 1.0,  pT6[2] = 0.78
pT7[0] = 1.0, pT7[1] = 0.75, pT7[2] = 0.63

Solution Cost[0] = -0.8518
Solution Cost[1] = -0.3741
Solution Cost[2..∞] = -1.2630

Overall Cost = -1,8449
```

Once again pT7 in final step is not relevant as any value of pT7 leads to the same result.

This clearly shows that more tests should be done in the beginning and that the probability of doing the tests should be reduced to minimize the false positives (as this cost is always incurred). It also shows that this more complex policy, with more decisions, leads to much better results than the simple decision one.

7 Conclusions and Future Work

This work shows how a fraud detection system can be modeled and analyzed in order to optimize the quality of its output. The approach used can easily be extended to more complex systems. Clearly, considering not only the tests dependency but also the test quality, results in better quality of fraud detection.

The initial FSA model presents some interesting characteristics that can be seen in real life fraud systems; the multiple testing stages and cascaded decisions. Although the analytical complexity of such a detection system may prevent a closed expression to be extracted it is always possible to numerically compute the values – even if the tests are not independent as was assumed in this work. The overall reward model is not complex (although it might be more complex than this).

As long as the staged test model is retained it would be easy to introduce a transfer matrix between stages. This could further increase the modeling power.

From the results it is also obvious that the fraud analysis history of a given customer is also relevant to optimize the tests to be performed. This clearly shows that the very common practice of applying the same test battery for customers regardless of their previous test history is not advisable.

Future work should consider correlated test results, test execution time (as this may also be a constraint) and variations to the overall cost and reward function.

The inclusion of execution time constraints is also relevant as this may impact the decision at each step. This is motivated by the fact that all customers should be checked within a detection cycle and some tests may require more time or processing power than is available.

Changes to the overall fraud detection process should also handle the possibility of a customer becoming a fraudster not at the beginning but later on (maybe due to theft of the mobile phone or financial difficulties of the customer itself).

References

[1] Bertsekas, Dimitri P. (1995), Dynamic Programming and Optimal Control. Vol 1. Athena Scientific.

[2] Cassandras, Christos G and Stéphane Lafortune (1999). Introduction to Discrete Event Systems. Kluwer Academic Publishers. Boston.

[3] Russell, J. and Norvig, P. (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, NJ: Prentice Hall, pp. 111-114

[4] Viswanadham, N. and Narahari, Y.(1992), Performance Modeling of Automated Manufacturing Systems. Information and System Sciences Series. Prentice Hall.